The STAR System:
A Unified Multi-Agent Simulation Model of Structure, Task, Agent, and Resource

David J. Kaplan
Carnegie Mellon University
djk@cmu.edu

February 24, 1999

CASOS Working Paper

This work was supported in part by the NSF IRI9633 662 and by the NSF GRT9354995 and by the ONR N00014-97-1-0037.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
24 FEB 1999		**00-00-1999 to 00-00-1999**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
The STAR System: A Unified Multi-Agent Simulation Model of Structure, Task, Agent, and Resource	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,Institute for Software Research, International,Center for Computational Analysis of Social and Organizational Systems,Pittsburgh,PA,15213**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**58**	

Abstract

Managers are continually designing and redesigning their teams and organizations. Design decisions tend to be based on trial and error, with little attention to long term experience and little or no attempt at verification. Organizational researchers interested in design have generated a vast compendium of design knowledge, much of which goes under the heading – Contingency Theory. As the name implies, the right design for an organization is seen to be contingent on a large number of complex and interacting factors. The complexity of the findings is such that, on the practical side, little guidance can be given to the manager and, on the theoretical side, advances in understanding are hampered by the overwhelming complexity. Computational models are ideally suited for reasoning about large complex systems composed of multiple interacting parts. This thesis addresses these pragmatic and theoretical problems by developing a computational toolkit for reasoning about organizational design that can be used to design teams or organizations, examine the impact of design changes, and reason theoretically about organizational design. The proposed toolkit, referred to as the STAR (Structure, Task, Agent, Resource) system, integrates a set of building block modules into a flexible agent oriented framework that can be used to model and simulate organizations and evaluate alternative designs. The development of this toolkit will advance both our theoretical understanding of organizations and our methodologies for reasoning about, and evaluating different, designs. Using this toolkit I will demonstrate the utility of a new theoretical proposition, permeability, and the benefit of a new method for engineering organizational designs, organizational morphing.

Table of Contents

1 Introduction

1.1 What am I doing and why

Organizations are complex, computational, and adaptive entities. Consequently they are difficult to theorize about. Recently progress has been made using computational models. However, one of the great difficulties here is that existing software for modeling is unsuited for the rapid development of organizational models - particularly when each agent in the organization is modeled. Researchers find that they are often re-inventing the wheel and spending an inordinate amount of time on issues of I/O, communication, and data gathering. This thesis responds to the lack of integrated multi-agent simulation toolkits for rapid organizational design.

For this thesis, I will develop a prototype of such a toolkit, use it to illustrate the power and limitations of conceptualizing organizations in terms of permeable agents, tasks, resources, and the relations among these permeable entities. This toolkit will provide a feature-rich test bed for organizational modelers and a set of exemplar models that users can alter or use at will. Using this toolkit I will demonstrate the flexibility of the toolkit by rapidly developing 3 alternative organizational models based on radically different types of tasks (search, classification, and configuration). I will demonstrate the power of the permeability approach to organization theory by showing how the same organizational performance can be achieved for the same task with different sets of agents, tasks, and structures. I will demonstrate the value of this approach for engineering new organizational design by using morphing techniques to determine redesign paths to achieve desired managerial goals.

1.2 The Need for a Modeling Toolkit

Organizations are performing major reorganizations of their divisions and employees. These reorganizations are often designed by experienced professionals: consultants outside the company or executives currently employed by the company. The reorganizations are complex and promise either increased performance or a return to profitability. However, these reorganizations are usually performed in an ad-hoc manner. While experts are utilized, those experts do not have access to tools that allow them to systematically explore alternative reorganization plans.

Corporations are spending hundreds of millions in reorganizations. In the second quarter of 1997 Apple Computer reported one time restructuring charges of $179 million (cnnfn.com, May 13, 1997). In January 1998, Black and Decker reported laying off 10% of its workforce (3,000 workers) and taking a restructuring charge of $300 million (cnnfn.com, January 27, 1998). 3M, in August 1998, announced 4,500 employees would be laid off and an associated $500 million charge would be taken for restructuring (cnnfn.com, August 27, 1998). Finally, Rockwell International announced a $625 million charge for restructuring (cnnfn.com, September 14, 1998).

Many theories exist for restructuring or reorganizing a corporation. In response to a query on the terms “corporate restructuring” Amazon.com provides 32 books and Borders.com lists 26 books. Newspapers and industry journals are rife with suggestions of how to reorganize. One author, discussing his book “The Creative Priority,” suggests that *creativity* is key in organizations and states that “…if the book has any value at all, it’s that it’s not coming from academic theory” (Evarts, 1998). In Supervision, a quote is derived from Hammer and Champy’s book “Re-engineering the Corporation” that

suggests that "…radical changes should be made to existing business practices" (Woolsey, 1997). It has been reported that the chairman of Mercedes-Benz, Juergen Schempp, "repeatedly stood conventional wisdom on its head, generally coming out a winner…" (Palmer, 1998). And finally, in Organizational Dynamics, the authors argue that "[m]any [management] techniques truly deserve the pejorative label, 'fad', and deserve to be strenuously combated" (Donaldson and Hilmar, 1998). These and other "techniques" along with academic theories overwhelm managers seeking to reorganize. Each author provides convincing arguments, and, often, a case study (perhaps the organization they "turned-around") to back up their ideas.

Efforts are underway in the research community to develop tools that aid the manager in doing redesign. For example, VDT was developed at Stanford by Levitt, et al., as a system for designing and evaluating design teams (Levitt, et al., 1994). These teams are responsible for design projects such as hospital buildings and petroleum refinery plants. Such tasks are often multi-year, consume millions of dollars, and often require over 100 employees (managers, engineers, etc.). These design efforts are complex and can be difficult to manage. VDT "…could provide accurate predictions for routine, fast-paced project organizations in which all participants were assumed to have congruent goals" (Thomsen, Levitt, and Kunz, 1998).

As another example, Milind Tambe's work on teamwork focuses on the real-world domain of fighter aircraft (Tambe, 1997). His work is based on the *Joint Intentions Theory* of Cohen and Levesque (Cohen and Levesque, 1990). The work is built on top of Soar (Newell, 1972) and Soar-IFOR (Tambe, Rosenbloom, and Schwamb, 1995). Humans interacting with his simulation report that the computer approximates the human

very well in simulated combat situations. Hence, simulated combat training now requires only one-half the human capital with his work.

Increasingly in basic research computational models of teams, groups, and organizations are being used to examine and discover new methods of coordination and control, the impacts of learning, evolution, and technology on organizational performance. For example, ORGAHEAD models organizational change (Carley and Svoboda, 1996). In this work, Carley and Svoboda found that re-engineering for performance gains may lead to performance decreases instead. The loss of "lessons of experience" of agents re-assigned or laid-off explains this unintended phenomena.

Another example, TAEMS (Task Analysis, Environment Modeling, Simulation), is a tool for modeling complex tasks from a variety of perspectives to study coordination theories (Decker, 1995). Decker and Lesser report on the results of simulations of theoretical organizations testing two hypothesis (Decker and Lesser, 1993). The first looked at the effects on performance between agents who partially shared their local views and those who did not. The second tested performance on different levels of decomposability of a technology. Their results show that performance is decreased by sharing local views and increased by increased decomposability of technology.

Finally, work by March details a computational model to study the concepts of Exploration and Exploitation (March, 1991). Exploration deals with "search, variation, risk taking, experimentation, play, flexibility, discovery, and innovation," while exploitation deals with "refinement, choice, production, efficiency, selection, implementation, and execution." March asserts that these concepts can be considered at

various levels of modeling from agent to agent interaction upward to societal interactions. Results from his model reinforce conventional *folk wisdom* such as the "returns to fast learning are not all positive" and "competitive victory does not reliably go to the properly educated."

Model Type and Name	**Authors**
Emulative Models	
Virtual Design Team (VDT)	Jin and Levitt (1996)
TacAir-Soar	Tambe (1997)
COMIT	Kaplan and Carley (1998)
Exploration and Exploitation	March (1991)
Garbage Can Model	Cohen, March, and Olsen (1972)
AAIS	Masuch and LaPotin (1989)
HITOP-A	Majchrzak and Gasser (1992)
ACTION	Gasser and Majchrzak (1994)
Cultural Transmission	Harrison and Carrol (1991)
Intellective Models	
ORGAHEAD	Carley and Svoboda (1996)
TAEMS	Decker and Lesser (1993)
STEAM	Tambe (1997b)
SDML	Moss (1998)
Plural Soar	Carley, et al. (1992)
SWARM	Minar, Burkhart, Langton, Askenazi (1996)
Organizational Consultant	Baligh, Burton, Obel (1994)
CORP	Carley and Lin (1995)
Sugarscape	Epstein and Axtell (1997)
MACE	Gasser, et al. (1987)
CONSTRUCT	Carley (1990)

Table 1. List of models and authors for organizations

What both the larger emulative models (such as VDT and TacAir-Soar) and the smaller intellective models (such as ORGAHEAD and TAEMS) have in common is that they both use formal representations of the organization and the individuals within it, and simulation techniques to explain, predict, and understand organizational behavior. Increasingly the emulative and the intellective models are becoming more similar. Both at the applied and theoretic level developers are finding that they are starting to re-invent

or re-build things others have. The work at both the emulative and intellective levels has led to a plethora of tools for looking at issues of organizational design and performance. See Table 1 for a list of the more prominent tools and models.

2 State of Current Models

There are a number of difficulties with these models. First, no model is comprehensive in terms of the basic elements of organizational design. One of the difficulties of current organizational models is that they treat the structure, task, agent, and resource as discrete and completely segregable entities. The argument has been that we can just build good agent models, task models, structure models, independently and simply plug and place them to create organizational models. I would suggest that this is wrong. By implementing one type of model, the other's are often ignored as is any concept of the relationships between models. Hence, there is no concept of how to integrate the models. Further, the models have different levels of specificity, and, when implemented, are often created in different languages and environments. Finally, and most importantly, research on organizations suggests that not only are there complex interactions among structure, task, agent, and resource, but that treating these as segregable leads to errors of interpretation and prediction. The research in contingency theory supports this view. Thus, from a modeling standpoint an integrated architecture is called for. This point is further discussed under the notion of permeability.

Although many simulation environments exist, none provide a single method through which a researcher can integrate models of structure, task, agent, and resource. I would argue that this is because they have been taking this plug and place approach.

Consequently, these existing models are not conducive to the design of group or organizational level models of either human or software-agent based organizations. They require a singular, or limited, view of the features of each model, something I argue against. Hence, they do not enable the user (researcher or practitioner) to dynamically examine alternative configurations of the organization.

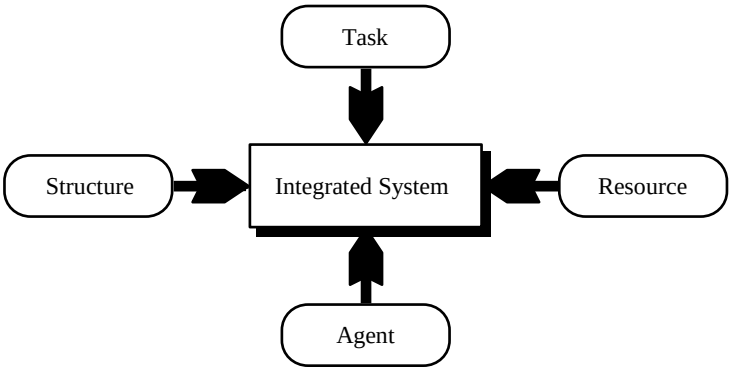


Figure 1. Unified model of organizations

To preview the theoretical analysis, if we think of organizations as being composed, at the most basic level, of structure, task, agent, and resource (Figure 1) then we find that most of the existing models leave out one or more of these elements (see Table 2).

	Agent		**Task**			**Structure**		**Resource**	
	Learning	*Multi-Agent*	*Types*	*Reassignment*	*Rework*	*Types*	*Dynamic*	*Kind*	*Amount*
VDT	❍	◗	❍	◗	●	❍	❍	◗	●
TAEMS	❍	●	●	❍	❍	◗	❍	●	●
STEAM	◗	●	◗	❍	❍	❍	◗	◗	●
COMIT	●	◗	◗	◗	❍	❍	❍	●	●
TacAir-Soar	◗	❍	❍	❍	❍	❍	❍	❍	❍
ORGAHEAD	●	◗	❍	●	❍	●	●	◗	●
Garbage Can	❍	◗	❍	❍	❍	◗	❍	❍	●

Table 2. Features of Component Models. ● - high, ◗ - medium, and ❍ - low

Second, models are written in different languages and are difficult to dock against each other (Axtell, et al., 1996). Docking is a process for comparing the output of models

whose purposes are claimed to be equivalent. It is important to be able to compare different models purporting to model the same behavior. Models are often implemented at different levels of specificity and represent variables in different ways. In order to compare the models, the models (simulation code) may need to be changed. These models are frequently written in different languages, reside on different platforms, and in many cases are modeled in different simulation systems. As a result, researchers seeking to dock multiple unfamiliar models, or even theirs with a similar one, have a daunting re-coding effort.

Finally, existing systems are somewhat inflexible. Existing models make it difficult, if not impossible, to perform what should be relatively simple, important changes to a modeled organization. The concept of flexibility is of great importance. Models that are *flexible* allow for relatively easy modification to support alternate hypothesis (i.e., for what-if analysis). Some systems are limited in a quantity they can represent (e.g., TacAir-Soar's maximum of 4 agents), others are limited in the breadth of modeling (e.g., ORGAHEAD has no model of technology (Carley 1996; Carley 1998)). Changing these models to allow for such concepts often requires rewriting of code (e.g., C, C++, or LISP). With these limitations, the systems are useful, but nonetheless are somewhat inflexible.

2.1 Toolkit Development as Theory Development

Developing a toolkit, the unified system, is equivalent to developing a theory in two ways – representation and hypothesis generation. Within cognitive psychology, decades of research on human cognition culminated in the development of unified architectures of cognition such as Soar and ACT-R. The findings embedded had a very general nature

such as – human cognition requires these components, or people use whatever information is salient to make decisions. These architectures embody aspects of these empirical findings in the form of a toolkit for representing and theorizing about human reasoning. These toolkits or architectures can be used to address reasoning in specific settings or about specific tasks by the researcher by adding in task specific knowledge, constraints, etc. Similarly, within organization theory and sociology, decades of research on groups, teams, and organizations have led to a body of findings that can be embedded in an integrated toolkit or system. Findings such as organizations are composed of certain components, and organizational decisions are affected by who takes part in the decision can be woven together to form a general architecture. The intent with STAR, is to do for organizational theory what Soar and Act-R did for cognitive psychology. Provide an initial toolkit that integrates known factors into a unified theory of organizational behavior.

First, STAR is a unified theory of organizational form as it specifies a particular representation of the organization in terms of the structure, task, agent, and resources. The relationships between models (i.e., structure) is similar to the PCANS model (Krakhardt and Carley, 1998), but I have created some specific extensions as described below. Although others have created models of task, agent, and resource, few have considered these models in the context of their relationships with the other models.

Second, given the toolkit, researchers are able to rapidly generate hypotheses. This thesis takes the stance that many other authors have. Namely that models, simulations, and theory are one in the same. Newell, in his William James Lectures at Harvard, emphasized that simulations are theory (Newell, 1990). He stated that "…theories tend

to be cast in terms of calculi, proof schemes, and simulations" and "[a] unified theory of cognition does not mean a high-level theory that gains its unification because it backs off to vagueness and platitudes. Calculations and simulations count" (Newell, 1990). Some authors use the term *simulation model* (Sterman 1996; Doran and Gilbert, 1994; Cyert 1994). Their point is not to somehow extend the concept of simulation to that of a model, but rather to make the point that a simulation is a model.

2.1.1 Theoretical Claims

This thesis builds out of and extends a structural approach to organizations. This approach argues that performance is affected not just by people or tasks or resources but by the structure of the relations among those entities and the number of those entities. While structural theory has a long tradition in organizational theory (Burns and Stalker, 1961; Blau and Scott, 1962; Mintzberg, 1979) this work builds directly on and extends a recent formalization of this body of research developed by Krackhardt and Carley – the PCANS model (Krakhardt and Carley, 1998).

According to the PCANS model the organization can be described by 3 entities: Individuals (i.e., agents), Tasks, and Resources. These entities are related through five relations (i.e., structures). Figure 2 lists the five relationships. The authors suggest the model is a base from which to build on. This thesis does just that.

PCANS currently treats the task as the lowest level of work. The limitation here is an implicit assumption that any agent assigned to a task performs all the work of the task. A concept offered in this thesis is that a *capability* is the lowest level of work. A task requires certain capabilities to be performed. Individuals (agents) have a set of

capabilities they can perform. Finally agents have a set of capabilities they must perform at each task. Capabilities extend the PCANS model with a new entity and two new relations allowing for more complete modeling of tasks.

PCANS

1. Precedence – temporal ordering of *tasks* (task-task);
2. Commitment of resources – mapping of resources to tasks that require them for completion (task-resource);
3. Assignment of personnel to tasks – mapping of individuals to the tasks they should perform (agent-task);
4. Network – relationships between individuals (agent-agent);
5. Skill – mapping of resources to an agent (agent-resource);

STAR additions

6. Agent Requirements – capabilities to be completed by an agent for a specific task (agent-capability-task); and
7. Task Requirements – capabilities to be completed to consider a task to be done or performed (task-capability).

Figure 2. Five Relations of the PCANS Model with additions for the STAR model

Research on organizational modeling suggests that a comprehensive and useful model of an organization should have an agent, task, organizational structure, and technology component (ACTS, Carley and Prietula 1995). Although models take into account one of these elements, as we saw earlier they do not take all into account. For example, VDT (Levitt, et al., 1996), ORGAHEAD, STEAM, and COMIT (Kaplan and Carley, 1998) model agent, task, technology, and structure, but do so separately. Decker's TAEMS provides an extensive task model while TAS and STEAM have both developed extensive agent models. ORGAHEAD provides a mechanism for specifying structure and seeks a

best structure. Finally, Levitt, et al., specify task using PERT-like semantics, information used by TAEMS. None of these models have extensive models of technology, and only STEAM and TAS have the ability to easily expand given their basis in Soar. Since each of these component model (TAMES, STEAM, VDT, TAS, COMIT, and ORGAHEAD) separately provide pieces of the puzzle, the question is, can we develop an integrated toolkit for developing organization models more quickly and efficiently if we combine these separate approaches?

In this thesis I extend this theoretical conception in three ways. First, through the recognition that lower level of specification is needed for many real world problems. Second is the thesis that the boundaries of agents, tasks, and resources are permeable. And third, that organizations are not static entities, but rather change, or morph, over time.

2.1.2 Increased specification

A lower level of specificity is necessary in organizational modeling and this thesis provides mechanisms for that lower level of specificity. Rather than modeling work as a task, we suggest that agents have capabilities they can perform and tasks have capabilities that must be performed. We specifically model messages as a means to communicate information between agents rather than a more classic approach of assuming common or shared knowledge. Hence, a modeler must detail the task they are modeling, not abstract away its richness. By forcing a more full specification (i.e., not abstracting the details) a forced *explication* occurs which may lead to new discoveries about organizations.

Extant simulations generally abstract away the details of what is being modeled. For example, the Garbage Can model represents Decision Makers (agents) as rows in tables along with a few condition statements and calculations. Such abstraction, although needed for the limited computational environment at the time, leaves the simulation as little more than a piece of code. The *theory* of a Garbage Can model is gone. Extracting the theory from the simulation is nearly impossible. In a similar vein, many of the simulations today take the same approach, though computational requirements can affordably be met. Performing such abstraction, therefore, should not afford the simulation implementor the ability to refer to their simulation as theory — the theory is abstracted away. Hence, because the proposed system forces increased specification, the modeler or researcher is left with a theory rather than simply a piece of code.

2.1.3 Permeability

This thesis puts forth the notion that the boundaries between structure, task, agent, and resource are *permeable*. That is, features of tasks can be shifted into structures; features of agents can be shifted into tasks; and so forth. For example, a metal stamping task requires stamps to create the final product. One implementation might have the stamp as part of an agent. Another might place the stamp as a resource that can be assigned to different agents. As such, the stamp is permeable between the resource model and the agent model.

Thus the act of designing (or redesigning) an organization is an act of determining the effect of configuration given some set of structure, task, agent, and resource, and not (as has typically been assumed) an act of coordination or optimization. These two notions of permeable boundaries and configurability underlie the proposed toolkit.

Permeability, as described here, is static. For example, the stamp in the metal stamping example is modeled *a priori* to the simulation. A dynamic approach would allow the stamp to move between the two models during task execution. The dynamic approach will not be considered in the thesis.

2.1.4 Morphing

An emerging concept in organizational modeling is *morphing*. In STAR, morphing is used as a tool that enables the researcher to see how the organization can change subject to a set of constraints and given a particular goal. An existing approach to morphing organizational design allows task responsibility to change through the execution of a task (Perdu and Levis, 1998). Their approach requires an *a priori* specification of which agents can perform which tasks, and in what order the agents should be assigned to the tasks. Distributed decision making rules residing with the agents decide which agents perform which tasks through the execution of the task.

Let us consider an example of how the Perdu and Levis morphing procedure works. Imagine the case of *back-up* secretaries who can perform work when the primary secretary is busy or absent. Assume there are three secretaries, Mary, Sue, and Doug, and a task that needs to be performed, photocopying. All three secretaries can perform photocopying, but it is primarily Mary's responsibility. The potential task assignments exist *a priori*. If Mary is not available, the morphing procedure creates an ordering of who should do the task. For example, the order generated by the morphing might be Sue should do it, and if not Sue, than Doug. When the task photocopy becomes necessary, distributed decision rules determine who actually performs the task given the ordering given by the morphing process.

The Perdu and Levis approach is limited in that it only applies to morphing agent assignments, explicitly, and agent communication, implicitly. This thesis will extend this morphing model to task morphing organizations (that tasks can be replaced by other tasks). Then, within STAR, morphing will be available as a tool for examining potential avenues of organizational change given known agents, tasks, and resources.

2.2 Toolkit Development as Method Development

By building the toolkit, advances in the methods of multi-agent organization simulation will be made. New methods are needed to support the unification of models not previously combined in such a flexible way. New methods also grow out of the process (e.g., permeability and composite agents). Finally, existing methods (e.g., morphing) can be expanded on.

2.2.1 What does the user need to be able to do

The demands on an organizational modeler are non-trivial. They must specify the number, type, and parameters of agents, tasks, and resources. Then they must specify relationships between these items (e.g., communication structures or task ordering and subtasking). Finally, specifying when and with what frequency data gathering takes place is needed to support the modeler.

2.2.2 Composite Agents

The system should be suitably flexible to support not just agents, but agents interacting to appear as a single agent. When modelers design agents, they rarely have the time or foresight to predict, and therefore incorporate, all possible variations on their work. Composite agents would allow other's to build upon (to extend) existing agents with ease. Such support would allow modelers to know less of the details of other's work (i.e.,

other's implemented code representing an agent) in order to build on it. For example, a modeler who has developed a theory about quality of data communicated between agents should be able to easily see its impact on existing organizations running various (different) tasks produced by many different researchers. By supporting the concept of composite agents, the modeler could use other's work by building agents who vary data communicated between agents. Then the modeler would couple their agent with existing agents, run the simulation, and compare the output.

2.2.3 Permeability

This thesis asserts that permeable boundaries between structures, tasks, agents, and resources exist. This is in contrast to the firm boundaries previously assumed by most formal modelers. The STAR system will allow modelers to change representations of items permeably between models. As such, the system allows modelers to test assumptions about permeability, and therefore develop theory. Permeability has not yet been implemented. Further discussion of permeability as method development is left for the body of the thesis.

2.2.4 Organizational Morphing

The existing organizational morphing technology focuses on agent responsibilities. A new method of morphing is proposed in this paper. Namely, one that allows tasks to be replaced by other tasks. Morphing has not yet been implemented. Further discussion of morphing as method development is left for the body of the thesis.

3 Toward A Unified Model

3.1 A unified approach

The building of the DAI and CMOT fields has resulted primarily from systems of limited scope, with little interoperability. It is argued, often in the conclusion of both CMOT and DAI publications, that the system can be expanded to more agents or different task domains. The "why" of such a motivation is clear, but at some level these arguments beg the simple question: how? Currently, researchers, either individually or in teams, build upon their systems by first borrowing or creating theories, and then implementing the theories as additions to existing models.

One approach could be to just expand current models. Ad-hoc expansion of current models provides new and possibly more flexible models, but may only extend and complicate yet further expansion. Expansion of existing models may continue to focus only on specific domains, and hence further confine that model to that domain. Common grounding to a method of interoperability can provide a better environment for building more complex, integrated systems. Such an environment cannot simply combine current work, but must allow for new theories as well.

The simple existence of a unified model is something many modelers are calling for. In fact this was one of the major topics at the recent SIMSOC conference in Europe (Italy, 1997). There is a recognized need for an environment where modeling of agent, task, technology, and structure can occur at varying levels of specificity. Rather than an incremental approach to such a goal, I propose a leap toward that goal by providing a unified system of models of structure, task, agent, and resource. Such an approach should help save duplicate effort better spent on honing specific models.

A unified model will allow for exploration of more complex and interesting environments, enable flexibility in that exploration, reduce re-implementation, and provide a mechanism for sharing of modeled components. More complex environments are fostered by the flexibility of the environment proposed. Assumptions in the code are moved to the modeling level where researchers can change structure or adjust parameters to create new situations. The reduction in re-implementation follows from a flexible modeling environment drawn from the most widely accepted models. Finally, researchers will be able to share such things as implemented agents, task or resource descriptions, and structural relationships because they are no longer integrated together in one piece of code. Although the agent is a piece of code, while the structural relationships and resource descriptions are data, the agents are not integral with the structural relationships or resources. Rather, they are *separate* items that can be used by other researchers in other circumstances. There are two approaches to creating such a unified model: combine existing or build a new framework. Each will be discussed in turn.

3.2 Solution 1: Combine Existing Models

This thesis began as an attempt to simply combine models. The concept is clear: take the best of each model and integrate them into one system. This builds off the idea that work is reproduced each time we build a new simulation. Hence, rather than building an entirely new system and repeating what others have done, the new system would be a fusion of existing validated systems.

3.2.1 Potential Component Models

There exist a set of models for structure, task, agent, and resource. To build from component models, a selection of models must first be made. Prime models include TAEMS, VDT, TacAir-Soar, STEAM, and ORGAHEAD. These systems are state of the art having gone through the rigors of successful validation. There is no model, *per se*, for *resource*. Resources have been relegated to a feature of an agent, assumed away in abstraction, or left at non-computational theoretical levels. Each of the models described below, combined with a computational view of resources, could provide a component for an integrated model of organization theory.

3.2.1.1 Task Models

There are two task models that can be drawn from to build a unified system. The first is Decker's TAEMS model of task. TAEMS provides a formal model of task and explicitly avoids a complex model of agency in hopes others will expand on that. The second model, VDT, fits nicely with TAEMS as VDT specifies task in a PERT-like way. TAEMS uses data that can be specified similarly to that of a PERT chart. But the TAEMS model is more flexible than VDT to the extent that TAEMS can model many different types of tasks, while VDT can model only one type, design.

The existence of each of these models has contributed a great deal to organizational modeling, but both lack features of agent, structure, and resource. Modeling task alone ignores the other models and their important interactions. For example, neither supports morphing, nor has the capacity to do morphing. The weak agent model in VDT, and the weak agent Application Program Interface (API[1]) in TAEMS, means neither can easily

[1] An API is a standardized specification for interaction with a system.

accept new agents. Finally, while TAEMS has a flexible task framework, VDT cannot be easily expanded beyond its application domain.

3.2.1.1.1 TAEMS

Keith Decker, in his 1995 Dissertation proposed the TAEMS (Task Analysis, Environment Modeling, and Simulation) framework. TAEMS responds to DAI's finding that coordination between interdependencies of task activities relies not on finding a singular coordination mechanism, but rather that there exists no singular mechanism. Instead, task interdependencies, coordination between activities, are dependent on environment. Three primary features of Decker's task modeling stand out: hard versus soft task interrelationships, multiple levels of task abstraction, and viewpoint of task structure. Hard constraints forbid certain tasks to begin until previous tasks are completed. Soft constraints allow subsequent tasks to begin, but previous tasks may positively impact subsequent tasks' quality.

TAEMS represents the concept of task at multiple levels of abstraction. At the highest level of abstraction, the task group consists of all tasks that have computational interrelationships. At the next level, the task is either a subtask or an executable method. The executable method is the lowest level of abstraction representing an action an agent might perform (e.g., replace hard disk drive). Removing a hard disk drive might seem like a higher-level task, but TAEMS leaves it up to the user to decide what level of detail an executable method is defined. Hence agents modeled in macro level simulations such as VDT and micro level simulations such as COMIT can both be represented.

Finally, TAEMS represents task from three viewpoints: generative, objective, and subjective. The generative view is a statistical view of task structure that generates the objective and subjective views. The objective view describes how tasks are related and what tasks are possible as time progresses. The objective view does not have information about specific agents. Finally, the subjective view describes at what time portions of a task become available to agents. Hence, the generative view provides a statistical method to generate task-centric and agent-centric views.

TAEMS is both a modeling language and an implemented simulation. The implementation is in object oriented Common Lisp (CLOS), and its graphical component is supported on the TI Explorer and DEC Alpha systems.

3.2.1.1.2 VDT

The Virtual Design Team (VDT) models large-scale, industrial design organizations. VDT is based on contingency theory, but also contains the author's observations about collaboration in large, multidisciplinary work groups. It is based heavily on information processing in organizations. Models of agent, communication tools, task, and organizational structure are present.

Agents in VDT are modeled with in- and out-boxes, preferences for information handling, and a set of skills. The skills include task and team experience, attention rules, productivity varieties, and the ability to fail. Agents cannot learn. Communication between agents occurs via the in- and out-boxes, with queues associated with each.

Communication is handled via several different technologies. VDT supports face-to-face meetings, telephone, and email. Agents are pre-disposed to use certain types of

technology for their communication. Characteristics of communication tools include priorities, selected targets, and natural idioms.

A task in VDT is modeled having interdependence and content. Task interdependence allows for tasks that are pooled, sequential, or reciprocal. Task content allows for real task attributes, stochastic generation, and *a priori* assignment. Modeled tasks are very specific to a particular domain, with the actions taken within subtasks not modeled.

Organizational structure in VDT models decision-making and communication between agents through Supervise/Report-to relationships in control structures and Coordinates-with relationship for communication structure. The control structure defines to whom agents report when failure occurs by specifying a level of centralization.

VDT appears to have some of the components (agent, task, technology, and structure), but its models of these components are not as extensive or flexible as other systems. For example, its agents are simply in- and out-boxes with skills and ability to communicate; unlike the agents in COMIT, VDT agents cannot learn. The only task that can be handled in VDT is routine design tasks; whereas, TAEMS can handle many types of tasks. In terms of technology, VDT considers communication technologies only in terms of whether they allow group or one-to-one communication; whereas, in COMIT, bandwidth, cognitive load, interaction style, etc. are all also considered. Its primary contribution, then, is its concept of task specification, which might be compatible with TAEMS

3.2.1.2 Agent Models

Two models of agency provide a good grounding for an integrated model. The first is TacAir-Soar. As a model of agency, TacAir-Soar relies on a rule-based system, Soar,

and focuses on very small groups of agents in a specific domain. A second model, STEAM, also focuses on small groups in the Soar architecture, but is designed as a shell for application to multiple domains.

Other agent systems exist, but for the purposes of this work are not considered. Two prominent ones are Sugarscape (Epstein and Axtell, 1996) and SWARM (Minar, et al., 1996). Sugarscape models societies of agents being born into, living in, and dying in a two-dimensional landscape. The landscape possesses sugar, and in an extension, spice. Sugarscape agents follow rules to accumulate sugar by consuming sugar to move from one position to another. Various interesting results have been shown, such as migration through the simulation of seasons. In a similar vein, SWARM is a system capable of modeling agent societies. SWARM agents can restructure themselves. Both Sugarscape and SWARM are capable of reproducing societal behavior. Although both provide interesting implications, they are far too simple for generalizable application to a unified system.

3.2.1.2.1 TacAir-Soar

TacAir-Soar (TAS) is a system being developed for the simulation of tactical air combat to train pilots. The goal of this work is to create intelligent automated pilots that are nearly indistinguishable from human pilots. The constraints to building such a system are two-fold, top-down and bottom-up constraints.

The top-down constraints result from the application environment, namely the simulated battlefield environment. Intelligent Pilots (IP) in such an environment have eight constraints. First, knowledge should be represented and organized to facilitate goal- and

knowledge-driven behavior. Second, pilots must be able to coordinate actions and behavior such that they can react to changing situations, perform multiple, overlapping tasks, and plan future actions such as routes. Third, the pilots should be able to learn so that performance is increased within and between simulations. Fourth and fifth, realistic human-like and real-time performance are necessary to provide an adequate training environment. Sixth, agents should be general enough for re-use by differing scenarios. Seventh, agents must be able to reason about the space they are in and the effect of time on their actions. Finally, agents must be able to interface with other agents for coordination, communication, explanation, and to develop models of other aircraft primarily with respect to tracking their low level actions.

The bottom-up constraints resulted from building TAS in an existing architecture, Soar. These constraints are often helpful, and can complement the requirements of the top-down constraints. For example, Soar's goal/problem space hierarchy provide a good environment for representing knowledge and reasoning about it. Unfortunately, Soar's learning mechanism, chunking, does little to increase performance either within or between simulations.

The top-down constraints should be thought of more as desired capabilities, rather than constraints. All of the desired capabilities have not yet been fully realized. For example, TAS is capable of handling a maximum of four pilots, with no more than two per team. The generalizability of TAS is limited to flying two types of aircraft on three types of mission.

3.2.1.2.2 STEAM

STEAM (Shell for TEAM work) focuses on small group (eight or less) coordination and communication. STEAM represents a fundamental shift in agent modeling by expanding it into the formal modeling of teamwork. The departure here is a rejection of the inflexibility of *a priori* coordination plans typical of previous work, and the assertion that agents must have integrated teamwork capability. The STEAM model of teamwork is based on the Joint Intentions Theory (Levesque, Cohen, and Nunes, 1990) with extensions for: "(i) team synchronization to establish joint intentions; (ii) constructs for monitoring joint intentions and repair, and (iii) decision-theoretic communication selectivity (to pragmatically extend the joint intentions theory)" (Tambe 1997a).

A separation of team from agent allows agents to execute hierarchical reactive plans. STEAM provides mechanisms to coordinate individual agent actions through the use of goals. Specifically Joint Persistent Goals (JPG) are goals to be jointly achieved. The member of the team trying to achieve a goal must first agree that it is not yet achieved, then achieve it, then communicate that it is complete. To enforce coordination, agents are required to communicate their view of the status of goals before taking individual action. Such a method avoids a scenario where an attack leader decides an attack is unachievable and returns to base without notifying others in the team. In addition to providing mechanisms for coordination in "normal" scenarios, STEAM allows for the monitoring of teams and their repair when a member is no longer available.

STEAM is implemented in Soar and has been used in three domains, two are military tasks and one is a sports task. One military task is attacking an enemy wherein attack helicopters fly from one location to another to engage an enemy. The other military task

is a transportation task moving troops from one location to another using escort aircraft. Finally, the sports application is for RoboCup, a form of synthetic soccer (Kitano et al. 1995).

A benefit of using a shell such as STEAM is its ability for reuse. First, because the focus is on team, STEAM can accept varying types of agents (e.g., pilots vs. soccer players). Hence, agents from differing systems can be reused in STEAM. Second, STEAM rules themselves can be reused across differing agent types. 100% of the STEAM rules between military applications are reused. Only 40%, however, are reused in the RoboCup application.

3.2.1.3 Structure Models

The ORGAHEAD model provides the most extensive simulation system of organizational structure. It models organizations providing a mechanism to evaluate alternative structures.

Considering structure alone will not suffice in the modeling of organizations. Such models abstractly represent the other models (i.e., task, agent, and resource). Task is often represented along a dimension like difficulty of task. Resources may be considered in terms of access and an agent's ability to use them, but structure models do not represent the coordination needed for the movement of resources between agents.

3.2.1.3.1 ORGAHEAD

ORGAHEAD is a model of organizational change and adaptation. It recognizes that knowledge exists not just in agents, but also in the linkages between agents. ORGAHEAD shows that different or changing structures will have performance

implications on an organization, and that changes at different levels in the organization will have impact on other levels.

	AGENT-				TASK-		
	Agent	Capability	Task	Resource	Capability	Resource	Task
VDT	●		●	●		●	●
TAEMS	●		●	●		●	●
STEAM			●	●			●
COMIT	●		●	●		●	●
TacAir-Soar	●		●	●		●	●
ORGAHEAD	●		●				●

Table 3. Structures included in component models. ● - included

ORGAHEAD supports up to 45 agents in at most three levels. Structure in the organization is formed via two types of ties: reporting and resource access. These ties may change through the course of a simulation run through hiring, firing, re-assignment, and re-engineering. Re-assignment changes to whom agents report while re-engineering changes the task and to which agents it is assigned. Changes in organizational structure are made as a result of measuring organizational performance via the CORP model. CORP models the agent, task, and organizational structure, while ORGAHEAD adds a mechanism to adjust the structure and move toward better structures with each move. This mechanism is based on the heuristic approach of Simulated Annealing.

3.2.1.4 Resource Models

Pfeffer and Salancik in their landmark book identified resources as one of the fundamental attributes of organizations (Pfeffer and Salancik, 1978). They define resources quite broadly to include skills, knowledge, and technology. They argue that to understand organizational performance, change, and inter-organizational adaptation it is necessary to trace the location and flow of resources and the resource needs of various tasks.

Resource is a generic term for the skill, knowledge, or technology that can be differentially distributed across agents within an organization. Resources are typically differentiated by type, either at a macro level – skills, knowledge, technology – or at a more micro level – such as in managerial knowledge, technical knowledge, staff-support knowledge. Resources are also often characterized in terms of amount – e.g., the number or skills, or the level of training with a particular skill, or the number of a particular device (number of computers).

3.2.2 Feasibility of combining component models

Combining component models is a very daunting task. The concept here is to literally combine the code base of these models thereby reusing code and reducing re-producing other's work. Only ORGAHEAD and COMIT are written in the same language, C. VDT is implemented using two commercial software products: Knowledge Engineering Environment and SimKit discrete event simulation system. Such products are expensive. TacAir-Soar, in contrast relies, at the core on freely distributable software, Soar. But support for the system is waning, in fact Soar's birth place, Carnegie Mellon University, has discontinued all official support. To the point, at the technical code level, combining such systems would require a tremendous depth of knowledge and support staff.

Even if the coding difficulties are viewed as surmountable, the model level of the systems are not necessarily directly compatible. Each system uses different input parameters, different output parameters, and, in the middle, different number of models combined in different ways. Visualizing COMIT's output requires third party software to translate its log-file format. In contrast, TacAir-Soar visualizes data in real time in complex simulated worlds. ORGAHEAD and VDT allow for exploration of different types of

agent to agent structures. However, VDT provides a different set of parameters to describe those relationships than does ORGAHEAD. Hence, input and output parameters along with models are not necessarily compatible across models. For example, replacing VDT's communication model with another's would render VDT useless as that communication mechanism is relied upon by the rest of the VDT system.

3.3 Solution 2: STAR system

Another approach is to build a new system (or toolkit) that is based on concepts from existing systems. There are two levels at which toolkit components can be discussed. The first is the model level, and the second the feature level. The model level consists of the primary components of STAR – structure, task, agent, and resource The feature level consists of individual features such as breakpoints or GUI modeling.

3.3.1 Toolkit Description – Models

3.3.1.1 STRUCTURE

There are seven structure models in STAR. Each consists of a relationship between models including task, agent, resource, and capability (a feature of agents). Some structures are within models (e.g., agent-agent and task-task) while others are between models (e.g., agent-task and task-resource). The purpose of each structure is outlined below, and its relation to the PCANS model is noted.

1. Task-Task – defines both the task being modeled as well as subtasks. Each task-task structure specifies the entry (start) and exit (end) point for the task or subtask. Structures that can be assigned to tasks can also be assigned to task-task structures. This is the P or precedence structure in PCANS.
2. Task-Resource – defines the resources required to complete a task. This is the C or commitment of resources structure in PCANS.

3. Agent-Task – defines the assignment of agents to tasks. This is the A or assignment of personnel to tasks structure in PCANS.
4. Agent-Agent – defines communication and reporting relationships between agents. STAR provides agents access to this structure. Each agent-agent structure is assigned to a task to define the communication structure between agents during the execution of a task. This is the N or network structure in PCANS.
5. Agent-Resource – defines the resources an agent has access to. This is the S or skill structure in PCANS.
6. Agent-Capability – defines which capabilities an agent must perform. Each such structure is assigned to a task to define which capabilities an agent must perform when asked to perform a task. Note that this structure does not define which capabilities an agent can perform; such a relationship is defined within the model of an agent. This structure is not in PCANS.
7. Task-Capability – defines the capabilities required for a task's completion. Note that all capabilities do not have to be performed by the same agent. This structure is not in PCANS.

3.3.1.2 TASK

The task model in STAR is generalizable to many different types of tasks. The model provides for flexible modeling of decomposable tasks where task precedence can be specified. Flexibility of the task model is derived from its ability to handle many types of tasks, model subtasks, and provide parallel execution of tasks. For example, Figure 3 shows a representation of the Radar Task (Lin and Carley, 1995) in STAR. After the task **Distribute Signal**, both **Judge Make Decision** and **Worker Decision** can be performed. Tasks may contain subtasks, those subtasks may contain other subtasks, and so forth. For example, Figure 4 shows the two tasks of the **Worker Decision** subtask. Tasks that contain no subtasks have the structures agent-task and agent-capability assigned to them.

Agent-task defines the agents assigned to a task, and agent-capability defines the capabilities the agent must perform for that task. Tasks are considered either not started, not complete, or done. A task graph (task-task structure) defines the precedence ordering of the tasks. The precedence ordering is through the use of hard constraints, hence a task must be *done* before successor tasks begin. Task graphs can branch allowing for parallel execution of tasks.

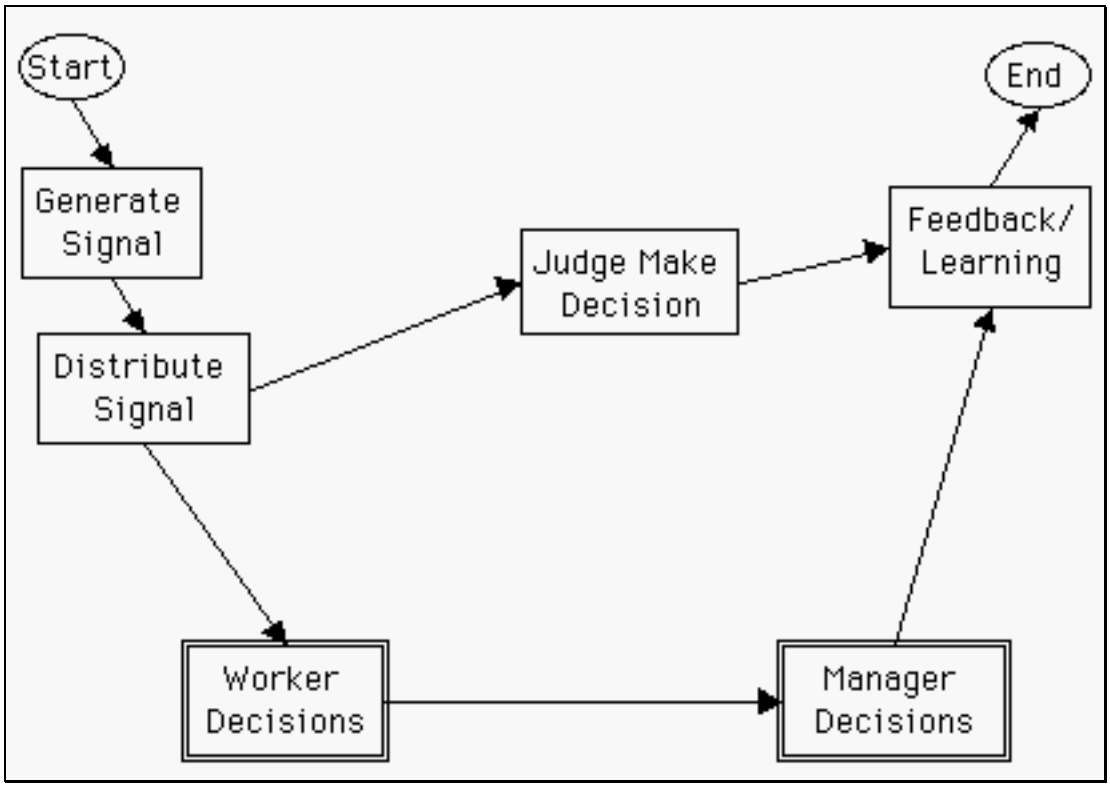


Figure 3. Radar Task Modeled in STAR

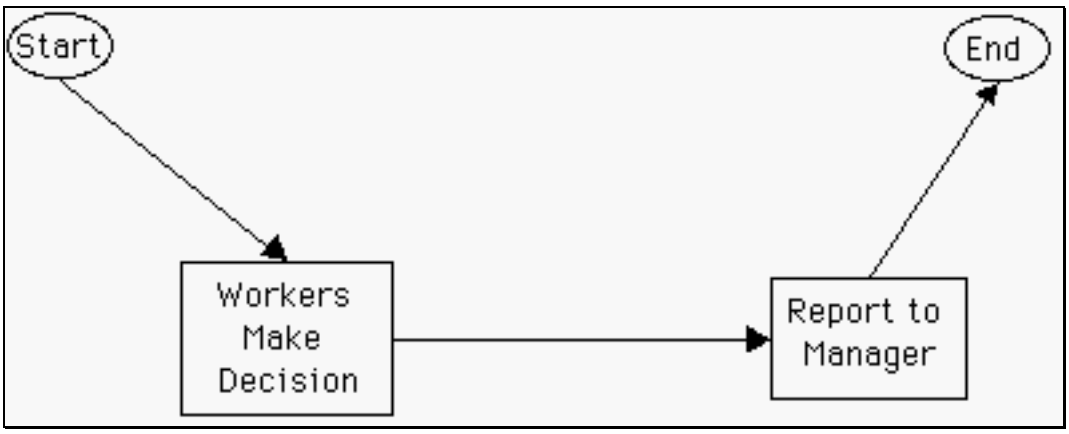


Figure 4. Radar Task Worker Decisions Subtask

3.3.1.3 AGENT

Agents can be compiled into STAR through an Application Program Interface (API) that defines the base model of a STAR agent. The only agent API that appears to exist for multi-agent organizational modeling is in TAEMS. It "appears" to exist only because no API has been published, but the creators claim others can create agents for TAEMS. The API in STAR requires that agents have in- and out-boxes for communication, perform work referred to as a capability, and provide access to agent parameters to STAR. The agent modeler can make decisions about learning mechanisms, communication content, skill level, and so forth. By providing a flexible interface for agent modelers, STAR allows for a wide range of agent types to be modeled.

In the Radar Task, for example, there are four types of agents. Two types perform the task, Workers and Managers, while the other two service the task, Judge and Signal Generator. Workers and managers coordinate their activities to aggregate a binary signal determining the signal's meaning. They perform the work. The other two agents do not perform the work of the task, but rather they service it. The signal generator generates a signal while the judge determines the signal's true meaning and provides feedback to all

agents as to the correct signal meaning. By allowing modelers to specify four different and distinct agents, it should be relatively easy to replace any agent. For example, if a researcher wishes to generate the signal differently, all one must do is write an agent to generate demand, compile that into STAR, and replace the old agent with the new one in the task specification.

The flexibility of plugging in agents to replace others extends to a concept we call Composite Agents. A composite agent is a grouping of two or more agents that act in concert as if they are one agent. Composite agents are meant to illustrate the power of being able to flexibly plug agents together. Composite agents require only two or more agents acting in concert appearing as one agent, there is no pre-defined arrangement or arrangements of agents. One example of composite agents might allow team members to interact as a team with other teams in an organization. Another example might allow researchers to easily test new theory by placing new agents between existing agents and the agent's they normally communicate with. In either case, an agent must be created, the Composite Manager Agent. Figure 5 illustrates a generic composite agent for the latter case, and Appendix A contains a detailed example of the Composite Agent concept. Both are described briefly below.

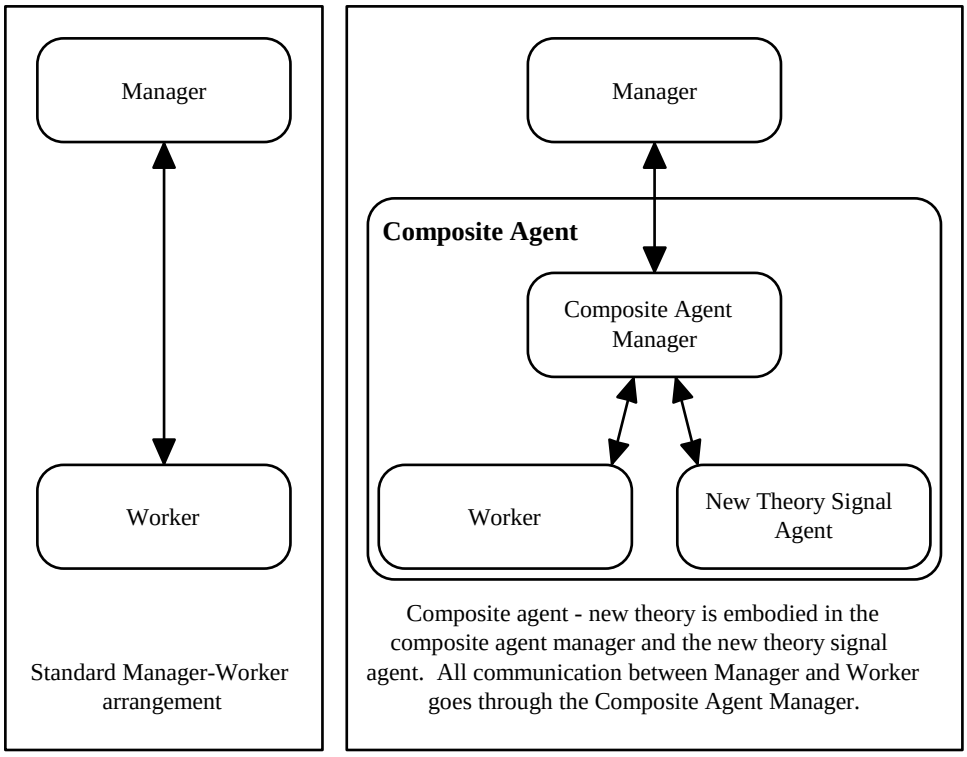


Figure 5. An illustration of a composite agent

The left side of figure 5 shows a manager and a worker who can communicate with each other on some task. To the right is a generic example of a composite agent. The original worker and manager still exist, but communication between the two go through a composite agent manager. The composite agent manager takes communication from the new theory signal agent, combines it in some way with the normal communication between the manager and agent, and sends the result to the manager. The combining of the normal communication with the new signal theory requires that the composite agent manager and the new theory signal agent contain the theory the researcher is trying to test. This example is one way of creating a composite agent, there may be many other ways.

A specific example is illustrated in Appendix A and detailed here briefly. In the Radar Task the worker and manager were left unchanged but two additional agents were added. One called a Composite Coordination Agent (the Composite Agent Manager of figure 5) and another called a Cooperate/Defect agent (the New Theory Signal Agent of figure 5). The worker or manager agent as well as the cooperate/defect agent reports all decisions to the composite coordination agent. Based on these two signals the composite coordination agent passes on a decision to its manager. Again, no modification was needed to the worker or the manager. Only two new agents and modeling adjustments in STAR are required.

3.3.1.4 RESOURCE

STAR models resources using the features Transferable, Persistence, and Quantity. Transferable allows resources to either be assigned to one and only one agent, or be transferable between agents. Resources are either persistent or depletable. Depletable resources are consumed as they are used; persistent ones are not. Resources can have any quantity, a fixed quantity, or a range of quantities. Resources are used by agents who specify how those resources are depleted.

3.3.2 Toolkit Description – Features

3.3.2.1 BREAKPOINTS

STAR borrows heavily from a concept of debugging computer programs. When debugging code, programmers are able to stop an application at specific places. Similarly, STAR provides breakpoint control while an organization is being simulated. Hence a researcher can allow a simulation to run until a condition (or one of many specified conditions) occur. Breakpoints do not have to stop the simulation run. They

can also provide visual and/or audio feedback as well. Through the use of audio breakpoints, we introduce the concept of "listening to an organization." Currently audio breakpoints create a specific sound, but can be expanded to play any sound. Sounds might be sampled representing things like the stamping of metal, or created on-the-fly based on the values of agent attributes (e.g., high pitch vs. low pitch for a success or failure).

3.3.2.2 GUI MODELING

GUI, or Graphical User Interface, provides researchers an easier to use interface to organizational simulations. In many previous systems, at best, a researcher could expect a cryptic command line interface that was not terribly flexible and often required restarting the simulation environment when an error was made. In contrast, STAR uses a GUI interface to allow for the specification of not just the task model, but also features of agents, structures, resources, setting of break points, importing of data, and data logging. Such an interface is a powerful alternative to existing command line interfaces or, worse yet, editing code for simple changes.

3.3.2.3 REAL-TIME FEEDBACK

In STAR, the researcher watches the organization in action, as it performs a task. The researcher may stop and examine messages being passed between agents, status of the tasks, and agent parameter values. The researcher can then simply continue the simulation as if the halt did not occur. In contrast, the researcher can also watch all of this information update as the simulation is running, and change which information is being viewed as the simulation progresses. This data is presented clearly to the user in a structured environment.

3.3.2.4 STATISTICS

STAR currently incorporates some simple statistics about quantity of each model instance (e.g., number or tasks or agents). This will be extended to include commonly used statistics about the structures including connectivity, density, efficiency, and redundancy.

3.3.2.5 IMPORT/EXPORT

A great deal of existing organizational work is represented in a matrix format. These matrices represent the relationships between agents, tasks, and resources. They are the structure of the organization. Additionally, methods and tools exist to perform analysis on organizations represented by matrices. Hence, in STAR, a researcher can import and export structures as matrices.

Imported data can be mapped to existing model instances. For example, if a matrix representing an agent-task structure is imported, the researcher can map rows and columns in the matrix to existing agents and tasks. Similarly, exported data can be easily imported into other programs such as UCINET or KrackPlot for further analysis.

Imports and exports support two DL formats and one raw matrix format, and allow for labels to be embedded in the matrix or in the header.

3.3.2.6 LOGGING

The same information a researcher can view either at breakpoints or while the simulation is run can be saved to a file (logged) for examination or analysis outside of STAR. Data logging records the simulation number together with any set of agent parameter values the researcher specifies.

4 Flexibility

Existing simulation systems typically focus on a single task or single type of agent. In contrast, STAR provides flexibility along each of its core components: Structure, Task, Agent, and Resource. Additionally, flexibility exists through importing and exporting various organizational structures and performing data analysis within or external to STAR.

Seven different structures representing relationships between agents, tasks, and resources provide a great deal of flexibility. Different organizational structures such as hierarchy or markets may be modeled. Relationships between agent and task can be changed to represent different resource allocation schemes. Finally, assignment of agents to tasks can be easily changed as well. All of these changes are accommodated through a graphical interface, or can be done via importing of pre-specified matrices.

Task flexibility allows for different types of tasks to be modeled as well as the modeling of those tasks in different ways. I will model three different tasks. The first is the classic Garbage Can model of Cohen, March, and Olsen (Cohen, et al., 1972). The second is a classification task, the Radar Task, in which agents interpret a signal and determine its meaning (Carley and Lin, 1995). Lastly, the Job Shop Task is a composition task requiring an organization of agents to perform stamping activities of cards. While the Garbage Can model has no managers, both of the other tasks can use managers. The Job Shop task will be modified between a hierarchical production scheme and an assembly line production scheme (detailed in the Permeability chapter).

By providing an API for agents to be compiled into STAR along with a GUI interface to their parameters, researchers can plug in many different types of agents. Each of the tasks with which flexibility of task is demonstrated are different in complexity, ability, and learning. Further, those agents can be adjusted via parameters that can be set at run time. Given the appropriate agent, a researcher can quickly and easily change its default experience or knowledge level. Additionally, because agents can easily communicate via their in and out boxes, agents can be combined into a Composite agent as is demonstrated in the Radar Task.

5 Permeability

Permeability is a concept introduced through the use of STAR. Traditionally models of agent, task, and resource have been thought of as being quite separate. In STAR we show that there is no absolute boundary between such models. To demonstrate permeability, we will implement the Job Shop Task[2] in two variations. The Stamp used to perform work will be modeled at the agent level as part of the agent in one implementation. In the other, the stamp will be modeled as a resource used by an agent.

In the first model, the agents will have an *a priori* assignment of stamps they can perform. Each agent will have one and only one stamp that they can apply to a card requiring the stamp. The stamp is part of the Agent model.

In the second implementation, stamps will be modeled as resources that can be assigned to any agent. Coordinating stamp use will require communication between agents, something not required of the one stamp per agent implementation. The agents do not

model stamps per se, rather than model the possession of and ability to use stamps as well as a coordination protocol for their use.

6 Organizational Morphing

The morphing of organizations is proposed by Perdu and Levis as the ability of an organization to change structure through the execution of a task (Perdu and Levis, 1998). They use a table mapping agents to tasks along with distributed decision making rules to perform organizational morphing.

The cells of the table mapping agents to tasks contain rank ordering information. A value of one specifies the first assignment of an agent to a task. That agent is assigned the task. If that agent cannot perform the task, an agent with a cell value of two for the same task is the second agent to be assigned the task. Distributed decision making rules define how the agents use the agent-task mapping table to dynamically reassign tasks.

I propose to both integrate organizational morphing into STAR and extend it with task morphing. Integrating organizational morphing will require further specification of the existing structure mapping agents to capabilities. Unlike Perdu's framework, STAR defines the finest grain of work on a task to be the capability, not a task. Where Perdu's work refers to task, we map that to Capability in STAR. Additional communication between STAR and its agents will also need to be implemented so that agents can access needed mapping information. No modeling of reassignment rules need be implemented; it is the agent modeler's responsibility to use the agent-capability mapping as desired.

[2] The Job Shop Task is a composition task. Workers use stamps to stamp cards of varying stamp requirements while managers assign cards (and possibly stamps) to workers.

The extension of task morphing will map tasks to tasks in a new way. Existing task-task mapping is for the definition of precedence constraints. A new task-task relationship will be defined specifying the ability to replace tasks with other tasks. Hence, as either agents or resources become unavailable, tasks may be replaced by other tasks.

7 Thesis Outline

1. Background - contents of proposal
2. Flexibility
    2.1. Background
        2.1.1. What is flexibility?
        2.1.2. State of the art – why these systems are not flexible enough
    2.2. Demonstrations
        2.2.1. Job Shop Task – as hierarchy vs. as assembly line
        2.2.2. Garbage Can
        2.2.3. Radar Task – base model
3. Permeability
    3.1. Background
        3.1.1. Typical ways models have been viewed – as non-permeable
        3.1.2. Arguments for why boundaries between models is not fixed (permeable)
        3.1.3. Examples of permeability
            3.1.3.1. What is permeability
            3.1.3.2. Resource as a model vs. resource as an agent attribute
            3.1.3.3. Task auto assignment versus people choose task
    3.2. Demonstration
        3.2.1. The Job Shop task – stamp as feature of agent vs. stamp as a resource
4. Morphing
    4.1. Background
        4.1.1. Concept of morphing drawing analogy to images
        4.1.2. Morphing in the organizational context

4.1.3. Current state of the art in organization morphing – agent to task table with weights plus rules

4.1.4. Extension to organizational morphing – tasks being replaced with other tasks

4.2. In-depth description of two different morphing techniques

4.2.1. Perdu and Levis' agentXtask morphing

4.2.2. New technique of taskXtask morphing

4.3. Demonstration

4.3.1. AgentXAgent – no task selected yet

4.3.2. TaskXTask – no task selected yet

5. Composite Agents

5.1. Background

5.1.1. What is a composite agent

5.1.2. Basic building blocks supported

5.2. Demonstration - application to Radar Task - Cooperate / Defect agents

5.2.1. How it is done

5.2.1.1. Existing agent communications

5.2.1.2. Cooperate / Defect agent

5.2.1.3. Composite Coordination agent

5.2.1.4. How to integrate at the System level

5.3. Results

6. Discussion and conclusions

6.1. Scope of the work

6.2. Contributions – take from the proposal

6.3. Extensions to the work

6.3.1. More complete:

6.3.1.1. Composite agent support

6.3.1.2. Permeability support

6.3.2. Agents should be able to exist separately from interface for computational and sharing reasons

6.3.3. There will be more to add as the system evolves

6.4. Insights gained and how they could be applied to extensions

8 Summary and Contributions

The STAR system is being proposed as an integrated modeling and simulation system for multi-agent organizations. Its feature rich environment is both drawn from existing models and consists of new concepts introduced in this thesis. The set of base models (structure, task, agent, and resource) have – separately – received the greatest attention by organization theorists. By integrating them into a single system, this thesis provides a leap forward in organizational simulations for both human- and agent-based systems.

STAR's contributions are numerous, and can be separated into two categories: theoretical and methodological. The theoretical contributions consist of permeability, morphing, explication, and templates. The methodological contributions include extensibility of the system, the agent model, debugging an organization in action, and a focus on object relationships.

8.1 Theoretical Contributions

8.1.1 Expressive Power of STAR

STAR is an expressive system, far more than previous multi-agent organization systems. STAR is both a modeling and simulation toolkit for modeling at varying levels of detail that will support concepts of permeability and morphing. It will be validated through the modeling and simulation of three representative tasks: Garbage Can Model, Radar Task, and Job Shop Task.

8.1.2 Explication

Existing models of organizations generally do not require a detailed break down of the task being performed, agent(s) being modeled, resources consumed, and relationships

between task, agent, and resource. Some models may require one of these, but no other existing organizational system require all of them. While some may argue it is a bad thing, indeed too time consuming, to detail all these items, I argue there is a benefit to explicating details of the task. Namely, by visiting the details, a researcher can gain insight into the theoretical relationships and implications that may exist. Existing system assume away the details by being *clever* in how they model an organization. In contrast, STAR forces explication, and hence may lead to theoretical insights otherwise missed.

8.1.3 Permeability

Other researchers implicitly or explicitly make the supposition that boundaries between models of structure, task, agent, and resource are fixed; features of one model cannot be part of another model. With STAR, it will be shown that quite the opposite is true. That boundaries, in some instances, do not exist, and model features may be interchangeable between models.

8.1.4 Morphing

Morphing is new to the organizations field. Existing work proposes that by allowing task responsibility to change through the execution of a task, the organization has been morphed. The organization *looks different* at the end of performing a task, than at the start. The concept maps well to organizations as they do change in various ways. STAR will incorporate this concept and extend upon it. The extension will allow for task substitution.

8.2 Methodological Contributions

8.2.1 Extensibility

Most organizational systems make it very difficult to extend what is modeled beyond very simple feature adjustment. Plugging in new agents, making small changes in agents, switching between types of organizational forms, or changing the characteristics of resources is generally very difficult. STAR provides mechanisms to make modeled organizations easily extensible.

8.2.2 Agent Model

Agents are modeled as code that interacts with the STAR system through an API. Agent models can be implemented to allow for plug in and use in different tasks, and can be designed to operate with other agents to form a composite agent (i.e., a group of agents acting in concert as one agent).

8.2.3 Debugging

STAR borrows from the computer science field the concept of debugging a program, and applies it to debugging an organization. Because STAR additionally incorporates the ability to watch the organization in action, a modeler can easily observe task progress and set breakpoints for interesting or problematic conditions. The modeler is no longer limited to the more typical batch processing mode of many organizational systems which do not allow researchers to view organizations in action.

Classic computer science debugging is used by an implementor to verify the behavior of her code. The goal or purpose of the implemented program is clear, and its behavior is being verified. Debugging in STAR can be used in the same way. However, researchers using STAR can go beyond this mechanistic verification. Just as social scientist sit as

observers in real organizations to study and learn, so too can researchers observe organizations in STAR. Organizations in STAR certainly need to be verified against their intent, but organizations are complex entities. We simulate them precisely for this reason. The researcher using STAR is observing an organization and its dynamics during a simulation via debugging. Extant systems mostly provide summary statistics with little ability to learn why the results occurred.

8.2.4 Object Relations

Because existing systems have focused primarily on one model at a time, little attention has been given to the interaction between models. STAR, by its very nature, focuses far more on this interaction. This allows modelers to begin to explore how parametric changes to each model affects other models and, in turn, overall organizational performance.

8.3 Summary

The STAR modeling and simulation system is a huge step forward from traditional organizational modeling approaches. It will provide a flexible system, in fact a test bed for organizational theorists to explore and possibly develop new theory. STAR is a more realistic approach to organizational modeling as it can be reused for a very wide array of tasks, agents, and resource. At the managerial (i.e., corporate world) level, and at the academic level, there are no tools that provide such a wide array of features in so sophisticated a system. Researchers and managers alike can both model and simulate organizations performing different tasks. Such an approach reduces the guess work of organizational design reducing costs associated with failure.

9 References

Aldrich, H., "Centralization versus Decentralization in the Design of Human Service Delivery Systems: A Response to Gouldner's Lament."

Axtell, R., Axelrod, R., Epstein, J.M., and Cohen, M.D., "Aligning Simulation Models: A Case Study and Results", in *Computational and Mathematical Organization Theory*, Vol. 1, No. 2, pp. 123-141, 1996.

Balci, O., Validation, Verification, and Testing Techniques Throughout the Life Cycle of a Simulation Study. Annals of Operations Research 53: 121-173, 1994a.

Balci, O., Principles of Simulation Model Validation, Verification, and Testing. Department of Computer Science, Virginia Tech, Blacksburg, VA., 1994b.

Baligh, H. H., Burton, R. M., Obel, B., Validating an Expert System That Designs Organizations, in K.M. Carley and M.J. Prietula, eds., *Computational Organization Theory*, pp. 179-194, Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.

Blau, P.M. and Scott, W.R. "The Concept of Formal Organization", *Formal Organizations: A Comparative Approach*, Chandler Publishing, 1962.

Burns, T.B. and Stalker, G.M., "Mechanistic and Organic Systems", *The Management of Innovation*, London: Tavistock Publication, pp. 119-125, 1961.

Carley, K.M., "Organizational Adaptation," Annals of Operations Research, 75: 25-47, 1998.

Carley, K.M., Organizational Adaptation and Cognition, in *Cognitive Science Proceedings*, forthcoming.

Carley, K.M. and Prietula, M.J. (Eds.), Computational Organization Theory, Hillsdale, NJ: Lawrence Erlbaum Associates, 1994.

Carley, K., "Computational and Mathematical Organization Theory: Perspective and Directions," Computational and Mathematical Organization Theory , #1 Vol. 1, pp. 39-56, 1995.

Carley, K.M., "Efficiency in a Garbage Can: Implications for Crisis Management," in March, James, and Roger Weissinger-Baylon (eds.), Ambiguity and Command: Organizational Perspectives on Military Decision Making, London: Pitman Publishing Co., Ch. 9, 1986.

Carley, K.M. (panel member), Modeling Human and Organizational Behavior: Application to Military Simulations, Final Report, Richard W. Pew and Anne S. Mavavor, editors. National Research Council, National Academy Press, 1998.

Carley, K.M., "A Theory of Group Stability," *American Sociological Review* , 56(3): 331-354, 1991.

Carley, K.M., Kjaer-Hansen, J., Prietula, M., Newell, A., "Plural-Soar: A prolegomenon to Artificial Agents and Organizational Behavior", in M. Masuch and M. Warglien, eds., *Distributed Intelligence: Applications in Human Organization*, pp. 87-118, Elsevier Science Publications, Amsterdam, The Netherlands, 1992.

Carley, K.M. and Gasser, L., "Computational Organization Theory," in Weiss, Gerhard (ed.) Distributed Artificial Intelligence, Cambridge, MA: MIT Press, Ch. 7, forthcoming.

Carley, K.M. and Lee, J., Dynamic Organizations: Organizational Adaptation in a Changing Environment.Ch. 15 (pp. 267-295) in Joel Baum (Ed.) Advances in Strategic Management, Vol. 15, Disciplinary Roots of Strategic Management Research, JAI Press, 1998.

Carley , K.M. and Lin, Z., "Organizational Designs Suited to High Performance Under Stress", in *IEEE - Systems Man and Cybernetics*, Vol. 25, No. 1, pp. 221-230, 1995.

Carley, K.M. and Prietula, M.J., ACTS Theory: Extending the Model of Bounded Rationality, In Carley, K.M. and Prietula, M.J. (Eds.), Computational Organization Theory. Hillsdale, NJ: Lawrence Erlbaum Associates, 55-87, 1994.

Carley, K.M. and Prietula, M.J., "Plural-Soar: Towards the Development of a Cognitively Motivated Theory of Organizations," in Proceedings of the 1993 Coordination Theory and Collaboration Technology Workshop . Symposium conducted for the National Science Foundation, Washington, D.C., 1993.

Carley, K. M. and Svoboda , D. M., "Modeling Organizational Adaptation as a Simulated Annealing Process", *Sociological Methods and Research*, Vol. 25, No. 1, pp. 138-168, 1996.

Cohen, M.D., March, J.G., and Olsen, J.P., "A Garbage Can Model of Organizational Choice", in *Administrative Sciences Quarterly*, Vol. 17, No. 1, 1972.

cnnfn.com, "Rockwell sets $400M charge: Electronic controls company to take additional restructuring charge in 4Q", http://cnnfn.com/hotstories/companies/9809/14/rockwell/, September 14, 1998.

cnnfn.com, "3M plans 500 more layoffs: Firm sets $500M restructuring charge, may sell units due to Asia, strong dollar", http://cnnfn.com/hotstories/companies/9808/27/3m/, August 27, 1998.

cnnfn.com, "Black & Decker remodels: Hardware company will cut 3,000 workers, sell units in restructuring", http://europe.cnnfn.com/hotstories/companies/9801/27/black/index.htm, January 27, 1998.

cnnfn.com, “Apple to stay in red longer: Losses expected to continue into fourth quarter, SEC filing says”, http://europe.cnnfn.com/digitaljam/9705/13/apple/index.htm, May 13, 1997.

Davis, S.M. and Lawrence, P.R., “The Matrix Organization - Who Needs It?”

Decker, K.S., Environment Centered Analysis and Design of Coordination Mechanisms, Ph.D. Dissertation, University of Massachusetts CMP SCI Technical Report, May, 1995a.

Decker, K.S., TAEMS: A framework for analysis and design of coordination mechanisms. In G. O’Hare and N. Jennings (Eds.), Foundations of Distributed Artificial Intelligence. Wiley Inter-Science, 1995b.

Decker, K.S., Task Environment Centered Simulation, In Prietula, M.J., Carley, K.M., and Gasser, L. (Eds.), Simulating Organizations: Computational Models of Institutions and Groups, AAAI Press/MIT Press, 1997.

Decker, K.S. and Lesser, Victor R., “Quantitative Modeling of Complex Environments”, *Intelligent Systems in Accounting, Finance and Management*, Vol. 2: 215-234, 1993

Donaldson, L. and Hilmar, F.G., “Management redeemed: The case against fads that harm management”, in *Organizational Dynamics*, Spring 1998.

Epstein, J.M. and Axtell, R., *Growing Artificial Societies*, MIT Press, Cambridge, MA, 1997.

Evarts, E.C., “Management’s Role in Fostering Creativity”, in *Christian Science Monitor*, August 6, 1998.

Freeman, L., Uncovering Organizational Hierarchy. Computational and Mathematical Organization Theory, 3(1):-5-18, 1997.

Galbraith, Jay, “Information Processing Model”, in *Designing Complex Organizations*, 1973.

Gasser, L. and Majchrzak, A., ACTION Integrates Manufacturing Strategy, Design, and Planning, in P. Kidd and W. Karwowski, eds., *Ergonomics of Hybrid Automated Systems IV*, IOS Press, Netherlands, 1994.

Harrison, J.R. and Carrol, G.R., “Keeping the Faith: A Model of Cultural Transmission in Formal Organizations, *Administrative Sciences Quarterly*, 36: 552-582, 1991.

Jin, Y. and Levitt, R.E., The Virtual Design Team: A Computational Model of Project Organizations, *In The Journal of Computational and Mathematical Organization Theory*, Vol. 2, No. 3, 1996.

Jones, R.M., Tambe, M., Laird, J.E., and Rosenbloom, P.S., (1994), Intelligent Automated Agents for Flight Training Simulators, In Proceedings of the Third Conference on Computer generated Forces and Behavioral Representation, Orlando, FL, pp. 33-42.

Kaplan, D.J. and Carley K.M., An Approach to Modeling Communication and Information Technology in Organizations, In Prietula, M.J., Carley, K.M., and Gasser, L. (Eds.), Simulating Organizations: Computational Models of Institutions and Groups, AAAI Press, 1998.

Kaufer, D.S. and Carley, K.M., Communication at a Distance: The Effect of Print on Socio-Cultural Organization and Change, Hillsdale, NJ: Lawrence Erlbaum Associates, 1993.

Krackhardt, D., "Graph Theoretical Dimensions of Informal Organizations.", in Carley, K.M. and M. Prietula (Eds.) Computational Organization Theory Hillsdale, NJ: Lawrence Earlbaum Associates, pp. 89-112, 1994.

Krackhardt, D. and Carley, K.M, "A PCANS Model of Structure in Organization" in Proceedings of the 1998 International Symposium on Command and Control Research and Technology, June, Monterey, CA, 1998.

Kitano, H., Asada, M., Kuniyoshi, Y., Nodi, I, and Osawa, E. (1995), RoboCup: The robot world cup initiative. In Proceedings of IJCAI-95 Workshop of the National Conference on Artificial Intelligence. Menlo Park, California: AAAI press.

Landry, M, Malouin, J.L., and Oral, M., Model Validation in Operations Research. European Journal of Operational Research, 14, pp. 207-220, 1983.

Levesque, H. J., Cohen, P. R., and Nunes, J., (1990), On Acting Together. In Proceedings of the National Conference on Artificial Intelligence. Menlo Park, California: AAAI Press.

Levitt, R.E., Cohen, G.P., Kunz, J.C., Nass, C.I., Christiansen, T., and Jin, Y., "The 'Virtual Design Team': Simulating How Organization Structure and Information Processing Tools Affect Team Performance", in K.M. Carley and M.J. Prietula, eds., *Computational Organization Theory*, pp. 1-18, Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.

Lin, Z., "A Theoretical Evaluation of Measures of Organizational Design: Interrelationship and Performance", in Carley, K. and M. Prietula (eds.) Computational Organization Theory , Hillsdale, NJ: Lawrence Earlbaum Associates, pp. 113-160, 1994.

Lin, Z. and Carley, K.M., "DYCORP: A Computational Framework for Examining Organizational Performance Under Dynamic Conditions," *The Journal of Mathematical Sociology*, 20(2-3): 193-217, 1995.

Majchrzak, A. and Gasser, L., "HITOP-A: A Tool to Facilitate Interdisciplinary Manufacturing Systems Design", in *International Journal of Human Factors in Manufacturing*, Vol. 2, No. 3, pp. 255-276, 1992.

March, J.G., "Exploration and Exploitation in Organizational Learning", in *Organizational Science*, Vol. 2, No. 1, February 1991

Masuch, M. and LaPotin, P., "Beyond Garbage Cans: An AI Model of Organizational Choice", in *Administrative Sciences Quarterly*, Vol. 34: 38-67, 1989.

Minar, N., Burkhart, R., Langton, C., Askenazi, M., "The SWARM Simulation System: A Toolkit for Building Multi-Agent Simulations", SantaFe Institute Working Paper, June 21, 1996.

Mintzberg, H., "The Five Basic Parts of the Organization", *The Structure of Organizations*, Prentice-Hall, Englewood, Cliffs, New Jersey, 1979.

Moss, S., "SDML: A Multi-Agent Language for Organizational Modeling", Journal of Computational and Mathematical Organization Theory, Vol. 4, No 1, pp. 43-79, 1998.

Newell, Allen, *Unified Theories of Cognition*, Harvard University Press, 1990.

Oral, M. and Kettani, O., The Facets of the Modeling and Validation Process in Operations Research. European Journal of Operational Research, 66 (No. 2), pp. 216-234, 1993.

Palmer, J., "Shake-up artist", in *Barron's*, March 23, 1998.

Perdu, D. M., A. H. Levis, "Adaptation as a Morphing Process: A Methodology for the Design and Evaluation of Adaptive Organizational Structures," *Computational & Mathematical Organization Theory*, Vol. 4, No. 1, pp. 5-41, 1998.

Pfeffer, J. and Salancik, G.R., The External Control of Organizations: A Resource Dependence Perspective, Harper & Row, New York, 1978.

Pidd, M.,. Tools for Thinking: Modeling in Management Science. John Wiley, Chechester, UK., Chapter 11, 1996.

Powell, W.W., "Neither Market Nor Hierarchy: Network Forms of Organization," in Staw, B. and L.L. Cummings (eds.), Research in Organizational Behavior, Vol. 2 JAI, 1998.

Prietula, M.J., Carley, K.M., and Gasser, L. (Eds.), Simulating Organizations: Computational Models of Institutions and Groups, Menlo Park, CA: AAAI Press / The MIT Press, 1998.

Robinson, S., Simulation Model Verification and Validation: Increasing the Users' Confidence. Proceeding of the 1997 Winter Simulation Conference (Andradcttir, S., Healy, K.J., Withers, D.H., and Nelson, B.L., eds.). The Society for Computer Simulation, San Diego, CA, pp. 53-59, 1997.

Rosenbloom, P.S., Johnson, W. L., Jones, R.M., Koss, F., Laird, J.E., Lehman, J.F., Rubinoff, R., Schwamb, K.B., and Tambe, M. (1994) Intelligent Automated Agents for Tactical Air Simulation: A Progress Report, In Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation, Orlando, FL.

SARGENT, R.G. (1996). Verifying and Validating Simulation Models. Proceedings of the 1996 Winter Simulation Conference (Charnes, J.M., Morrice, D.J., Brunner, D.T., and Swain, J.J., eds.). IEEE, Piscataway, NJ, pp. 55-64.

Tambe, M., Rosenbloom, P.S., Schwamb, K., Constraints and Design Choices in Building Intelligent Pilots for Simulated Aircraft: Extended Abstract, In AAAI Spring Symposium on "Lessons Learned from Implemented Software Architectures for Physical Agents," 1995.

Tambe, M., (1996) Teamwork in Real-world, Dynamic Environments, International conference on multi-agent systems (ICMAS96).

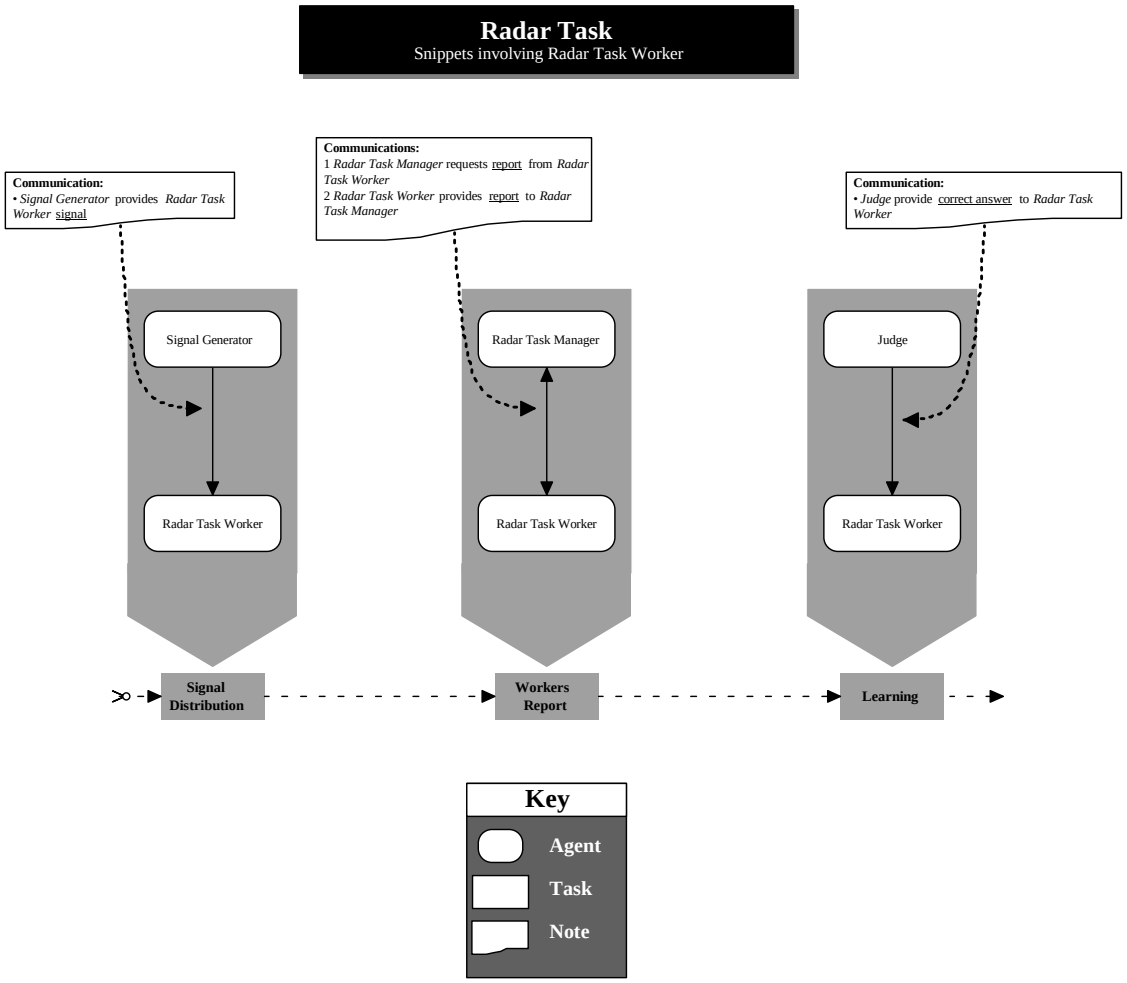
Tambe, M., "Agent Architectures for Flexible, Practical teamwork, in Proceedings of the AAAI American Association of Artificial Intelligence (AAAI), 1997a.

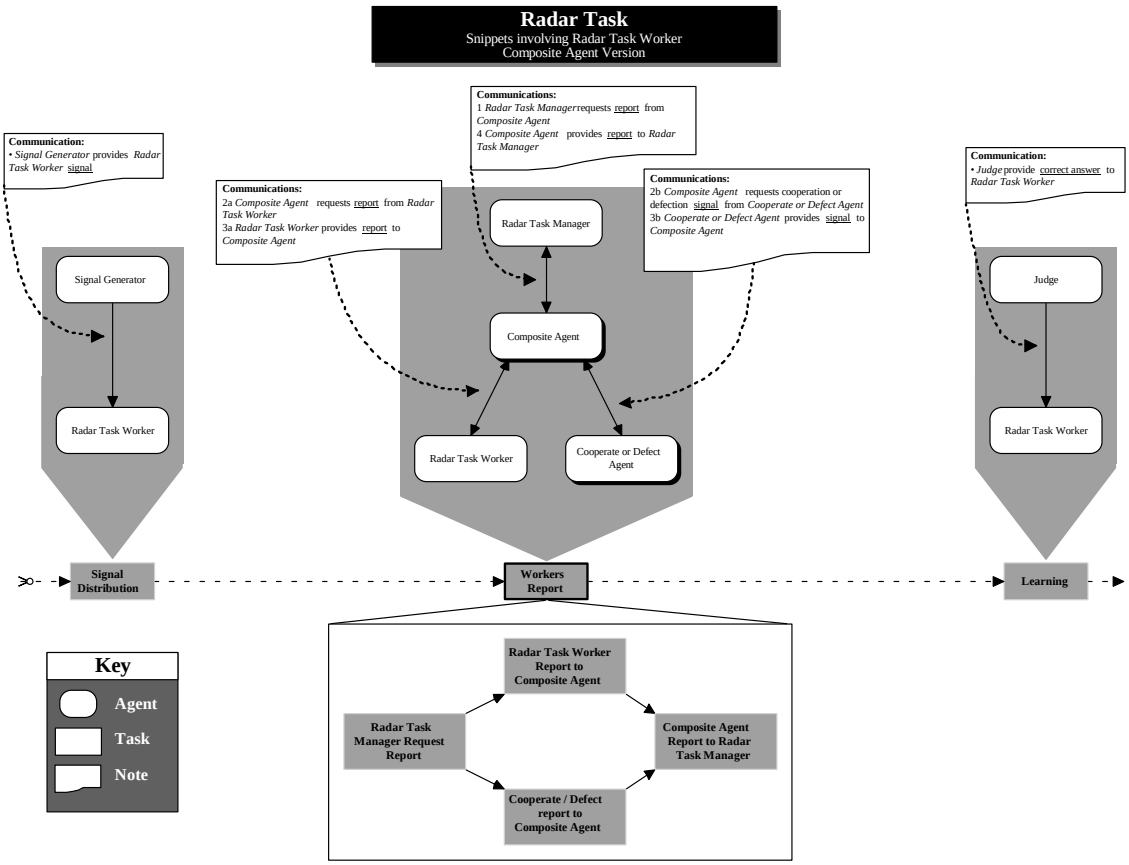
Tambe, M., (1997b) Towards Flexible Teamwork, In Journal of Artificial Intelligence Research, Vol. 7, Pages 83-124.

Thomsen, J., Levitt, R.E., and Kunz, J.C., "A Proposed Trajectory of Validation Experiments for Computational Emulation Models of Organizations," CIFE Working Paper 047, 1998.

Woolsey, R., "Building high trust organizations", in Supervision, September 1997.

Appendix A. Detailed Example of Composite Agent




Radar Task
Snippets involving Radar Task Worker
Composite Agent Version
Communication:
• Signal Generator provides Radar Task Worker signal
Communications:
1 Radar Task Manager requests report from Composite Agent
4 Composite Agent provides report to Radar Task Manager
Communications:
2a Composite Agent requests report from Radar Task Worker
3a Radar Task Worker provides report to Composite Agent
Communications:
2b Composite Agent requests cooperation or defection signal from Cooperate or Defect Agent
3b Cooperate or Defect Agent provides signal to Composite Agent
Communication:
• Judge provide correct answer to Radar Task Worker
Signal Generator
Radar Task Worker
Radar Task Manager
Composite Agent
Radar Task Worker
Cooperate or Defect Agent
Judge
Radar Task Worker
Signal Distribution
Workers Report
Learning
Key
Agent
Task
Note
Radar Task Worker Report to Composite Agent
Radar Task Manager Request Report
Composite Agent Report to Radar Task Manager
Cooperate / Defect report to Composite Agent